



UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
-----------------	-------------	----------------------	---------------------	------------------

10/693,659

10/24/2003

Jeffrey P. Snover

MS1-1741US

9647

22801 7590 03/19/2009

LEE & HAYES, PLLC
601 W. RIVERSIDE AVENUE
SUITE 1400
SPOKANE, WA 99201

EXAMINER

ABEL JALIL, NEVEEN

ART UNIT

PAPER NUMBER

2165

MAIL DATE

DELIVERY MODE

03/19/2009

PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

DETAILED ACTION

Remarks

1. In response to Applicant's Amendment filed on February 12, 2009, claims 1-11, and 13-23 are pending.
2. Applicant's amendment to the specification has been received and entered.
3. Applicant's amendment to the claims has overcome the previously presented claim objections.

Claim Rejections - 35 USC § 103

4. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

(a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negated by the manner in which the invention was made.

5. Claims 1-11, and 13-23 are rejected under 35 U.S.C. 103(a) as being unpatentable over Murray et al. (U.S. Pub. No. 2006/0235968 A1) in view of Young (U.S. Patent No. 6,782,531 B2) and further in view of Kirens (U.S. Patent No. 5,864,862)

As to claims 1, and 19, Murray et al. discloses a system that extends data types available to an operating environment, the system comprising:

a processor (See Murray et al. page 3, paragraph 0024); and

a memory, the memory being allocated for a plurality of computer-executable instructions which are loaded into the memory (See Murray et al. page 3, paragraph 0024) for execution by the processor, the computer-executable instructions comprising:

parsing a sequence of object-based commands into individual object-based commands (See Murray et al. page 7, paragraph 0090);

associating each individual object-based command with at least one execution element (See Murray et al. page 6, paragraph 0085).

Murray et al. does not teach executing each execution element associated with each individual object-based command to produce output objects, wherein the execution of each execution element is execution dependent upon an execution-supporting operating environment.

Whereas, in the same field of endeavor, Young discloses a data processing system including a transaction processor pipeline made up of a number of pipeline stage (sub-component) and a parser for parsing data object; executing each execution element associated with each individual object-based command to produce output objects, wherein the execution of each execution element is execution dependent upon an execution-supporting operating environment (FIG. 2-3; col. 7, lines 15-61).

It would be obvious to one having ordinary skill in the art at the time the invention was made to modify Murray et al.'s invention with Young's invention to include a pipeline objects executed in different stage as sub-component for data processing. One would have been motivated to provide include such feature in order to provide a flexible, distributed system for performing calculations defined (Young's col. 8, lines 48-55).

The combination of Murray et al. and Young still does not teach resolving each object-based command in the sequence of object-based commands to a data type; and

for data types that are not natively supported by the operating environment, retrieving extended information that defines the data types and creating an instance of the data types for each object-based command in the sequence that was resolved to one of the data types.

Kirens teaches resolving each object-based command in the sequence of object-based commands to a data type (See Kirens column 15, lines 34-36, and see column 12, lines 56-60, and column 16, lines 11-19); and

for data types that are not natively supported by the operating environment, retrieving extended information that defines the data types and creating an instance of the data types for each object-based command in the sequence that was resolved to one of the data types (See Kirens column 15, lines 21-26, and see column 15, lines 42-45, wherein "user defined types" are non-native to operating environment).

It would be obvious to one having ordinary skill in the art at the time the invention was made to further modify Murray et al. as modified with the teachings of Kirens's invention to include resolving each object-based command in the sequence of object-based commands to a data type; and for data types that are not natively supported by the operating environment, retrieving extended information that defines the data types and creating an instance of the data types for each object-based command in the sequence that was resolved to one of the data types

because it would introduce and accommodate further newly discovered data types relative efficiently to reduce the execution time.

As to claims 2, and 20, Murray et al. as modified discloses wherein the executing act comprises processing each execution element in order of each execution element's associated individual object-based commands in the sequence of object based commands (See Murray et al. page 6, paragraph 0087, and see Kirens column 20, lines 52-56).

As to claims 3, and 21, Murray et al. as modified discloses wherein the executing further comprises inputting into one or more execution elements output objects produced from one or more previously processed execution elements (See Murray et al. page 5, paragraph 0067, and see Murray et al. page 6, paragraph 0076, and see Young FIG. 2-3; Young col. 7, lines 15-61).

As to claims 4, 16, and 22, Murray et al. as modified discloses wherein the extended information comprises extended metadata and code, the extended metadata describes the data type and the code comprises additional instructions to populate the instance of the data type (See Murray et al. page 6, paragraph 0082, wherein "attributes" are read on "metadata").

As to claim 5, Murray et al. as modified discloses comprising receiving the sequence of object-based commands via an object-based command pipeline (See Young column 7, lines 16-44, and see Young column 8, lines 1-10), wherein the sequence of object-based commands includes a wildcard and the processing further comprises producing a subset of the sequence of object-based commands based on the wildcard (See Murray et al. page 5, paragraph 0069, wherein it is inherent in any command language that character representing wildcard is

Art Unit: 2165

accepted).

As to claim 6, Murray et al. as modified discloses further comprising receiving the sequence of object-based commands via an object-based command pipeline (See Young column 7, lines 16-44, and see Young column 8, lines 1-10), wherein the sequence of object-based commands includes a property set and the processing further comprises identifying a plurality of properties associated with the property set and processing the sequence of object-based commands based on the plurality of properties (See Murray et al. page 7, paragraph 0090).

As to claim 7, Murray et al. as modified discloses comprising receiving the sequence of object-based commands via an object-based command pipeline (See Young column 7, lines 16-44, and see Young column 8, lines 1-10), wherein the sequence of object-based commands includes a relation and the processing further comprises finding items that the sequence of object-based commands consume based on the relation (See Young column 7, lines 48-61, also see Murray et al. page 7, paragraph 0096, lines 16-20, wherein “sequential” is taught).

As to claim 8, Murray et al. as modified discloses further comprising receiving the sequence of object via an object-based command pipeline (See Young column 7, lines 16-44, and see Young column 8, lines 1-10), wherein the sequence of object-based commands comprises a property path, the property path comprises a series of components that provide navigation to a desired property of each object -based command in the sequence (See Murray et al. page 7, paragraph 0092, also see Murray et al. page 7, paragraph 0096, lines 16-20, wherein “sequential”

Art Unit: 2165

is taught).

As to claims 9, and 23, Murray et al. as modified discloses wherein the sequence of object-based commands is associated with a first data type and the processing further comprising looking up a conversion for converting the first data type to the data type (See Murray et al. page 5, paragraph 0067, and see Murray et al. page 6, paragraph 0076).

As to claim 10, Murray et al. as modified discloses wherein each component comprises a property of each object-based command in the sequence, a method of each object-based command in the sequence, a field of each object-based command in the sequence, a third party property, or a third party object method (See Young column 8, lines 48-64).

As to claim 11, Murray et al. as modified discloses wherein the sequence of object-based is received as input to a subsequent command in the object-based command pipeline after processing the sequence of object-based commands (See Young column 7, lines 16-44, and see Young column 8, lines 1-10).

As to claim 13, Murray et al. as modified discloses wherein a component comprises a reference to registered code (See Murray et al. page 4, paragraph 0055).

As to claim 14, Murray et al. discloses a computer storage medium for facilitating resolution of partially unresolved input, the medium having computer executable instructions, which when executed by a computer perform operations comprising:

receiving one or more parseable input objects (See Murray et al. page 7, paragraph 0090), the input objects being output from an already processed execution element that is associated with one or more object-based commands of a sequence of commands obtained within an execution-supporting operating environment, wherein the execution of an execution element is execution dependent upon the execution-supporting operating environment in order to actually execute (See Murray et al. page 6, paragraphs 0086-0087);

retrieving extended information that defines the data type (See Murray et al. page 6, paragraph 0076); and

creating an instance of the data type (See Murray et al. page 5, paragraph 0069), wherein the receiving, retrieving, and creating acts facilitate resolution of partially unresolved input.

Murray et al. discloses the claimed invention but does not explicitly teach via an object-based command pipeline.

Whereas, in the same field of endeavor, Younq discloses a data processing system including a transaction processor pipeline made up of a number of pipeline stage (sub-component) and a parser for parsing data object; and sending, with the method associated with a first pipeline sub-component, the at least one object to a next pipeline sub-component for processing, the next pipeline sub-component being one of the plurality of pipeline sub-components; and outputting information from at least one of the plurality of pipeline sub-

Art Unit: 2165

components, the information is the result of at least a portion of the processing performed by the object based pipeline (FIG. 2-3; col. 7, lines 15-61).

It would be obvious to one having ordinary skill in the art at the time the invention was made to modify Murray et al.'s invention with Young's invention to include a pipeline objects executed in different stage as sub-component for data processing. One would have been motivated to provide include such feature in order to provide a flexible, distributed system for performing calculations defined (Young's col. 8, lines 48-55).

The combination of Murray et al.'s invention with Young's still does not teach the one or more parseable input objects including content that uses a data type that is not natively supported by the execution-supporting operating environment.

Kirens teaches the one or more parseable input objects including content that uses a data type that is not natively supported by the execution-supporting operating environment (See Kirens column 15, lines 21-26, and see column 15, lines 42-45, wherein "user defined types" are non-native to operating environment).

It would be obvious to one having ordinary skill in the art at the time the invention was made to further modify Murray et al. as modified with the teachings of Kirens's invention to include the one or more parseable input objects including content that uses a data type that is not natively supported by the execution-supporting operating environment because it would introduce and accommodate further newly discovered data types relative efficiently to reduce the execution time.

Art Unit: 2165

As to claim 15, Murray et al. as modified discloses wherein the one or more parseable input objects comprises a Windows Management Instrumentation (WMI) input, an ActiveX Data Object (ADO) input, an XML input, or a third party data format (See Murray et al. page 6, paragraph 0077).

As to claim 15, Murray et al. as modified discloses wherein the one or more parseable input objects comprises a third party object that provides an additional property to an object supported natively within the execution-supporting operating environment (See Murray et al. page 5, paragraph 0064, and see Murray et al. page 6, paragraph 0079).

As to claim 18, Murray et al. as modified discloses wherein the one or more parseable input comprises an ontology service (See Murray et al. page 3, paragraph 0027, and see Murray et al. page 6, paragraph 0079, wherein “ontology” reads on “grammar”).

Response to Arguments

6. Applicant's arguments filed on February 12, 2009 have been fully considered but they are not persuasive.

Applicant's argument that “Murray fails to teach or suggest "parsing a sequence of object-based commands into individual object-based commands" is respectfully noted but not found to be persuasive.

Although the applicant correctly noted an introductory section of Murray paragraph 0013 that heavy reliance on OO programming suffers from execution overhead; hence, Murray went ahead and developed a method to continuing programming in OO by including of CLI (sequence of action in a single command interface, see paragraph 0090) leading to reduce such deficiencies in associated with OO. All while Murray is consistent in performing his invention within the realm of object oriented programming which attributes objections to each defined entity. More specifically, Murray as whole continues the use of object-oriented commands to access and process executions of managed object in a managed object database as shown in paragraphs 0055, 0063, 0065, and 0084-0085 (wherein the system as a whole includes applications accessing and servicing the managed object which are inherently using object code, and wherein each CLI is send to be executed at the corresponding managed object entity).

Applicant's arguments on page 20 appear to be directed more to the specification in highlighting the benefits of Cmdlets which are not claimed.

Indeed Murray's CLI 's are build from individual commands (actions) as acknowledged by the Applicant's remarks paragraph 0023, therefore, as they are run though the parser, firstly they are decomposed, matched to the dictionary, and then executed in sequence (every CLI is decomposed to the individual commands all build using object-oriented code) which is read on the argued limitation.

Conclusion

7. **THIS ACTION IS MADE FINAL.** Applicant is reminded of the extension of time policy as set forth in 37 CFR 1.136(a).

A shortened statutory period for reply to this final action is set to expire THREE MONTHS from the mailing date of this action. In the event a first reply is filed within TWO MONTHS of the mailing date of this final action and the advisory action is not mailed until after the end of the THREE-MONTH shortened statutory period, then the shortened statutory period will expire on the date the advisory action is mailed, and any extension fee pursuant to 37 CFR 1.136(a) will be calculated from the mailing date of the advisory action. In no event, however, will the statutory period for reply expire later than SIX MONTHS from the mailing date of this final action.

8. Any inquiry concerning this communication or earlier communications from the examiner should be directed to Neveen Abel-Jalil whose telephone number is 571-272-4074. The examiner can normally be reached on 8:30AM-5:30PM EST.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Christian Chace can be reached on 571-272-4190. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Art Unit: 2165

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see <http://pair-direct.uspto.gov>. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free). If you would like assistance from a USPTO Customer Service Representative or access to the automated information system, call 800-786-9199 (IN USA OR CANADA) or 571-272-1000.

Neveen Abel-Jalil
March 16, 2009
/Neveen Abel-Jalil/

Primary Examiner, Art Unit 2165